

特表平10-512699

(43)公表日 平成10年(1998)12月2日

(51)Int. Cl.⁶

G 0 6 F 17/30

識別記号

F I

G 0 6 F 15/40 3 1 0 F
15/419 3 2 0

審査請求

有

予備審査請求

有

(全38頁)

(21)出願番号 特願平9-522215
 (86)(22)出願日 平成8年(1996)12月10日
 (85)翻訳文提出日 平成10年(1998)3月24日
 (86)国際出願番号 PCT/US96/19831
 (87)国際公開番号 WO97/22069
 (87)国際公開日 平成9年(1997)6月19日
 (31)優先権主張番号 08/571,748
 (32)優先日 1995年12月13日
 (33)優先権主張国 米国 (U S)

(71)出願人 デジタル イクイブメント コーポレイ
 ション
 アメリカ合衆国 マサチューセッツ州 0
 1754 メイナード パウダーミル ロード
 111
 (72)発明者 モーニア ルイス エム
 アメリカ合衆国 カリフォルニア州 940
 61 レッドウッド シティー メリーラン
 ド ストリート 2019
 (74)代理人 弁理士 中村 稔 (外6名)

最終頁に続く

(54)【発明の名称】コンピュータのネットワークからワールドワイドウェブ上のページを捜し出したり、ドキュメントを捜し出したりするためのシステム及び方法

(57)【要約】

迅速に、ネットワークによって接続されているコンピュータからワールドワイドウェブ上のウェブページをフェッチして解析するためのウェブクローラシステム及び方法であり、ランダムアクセスメモリ (RAM) に記憶されたハッシュテーブル及びシーケンシャルウェブインフォメーションディスクファイルを含む。システムに既知である全てのウェブページについて、ウェブクローラシステムは、ハッシュテーブルにより小さなエントリを記憶するのに加えて、シーケンシャルディスクファイルにエントリを記憶する。ハッシュテーブルエントリは、識別値、対応するウェブページが上手くフェッチされた時だけ真がセットされるフェッチフラグ、対応するエントリがシーケンシャルディスクファイルのどこに記憶されているかを示すファイル位置インジケータを含む。シーケンシャルディスクファイルのエントリの各々は、対応するウェブページのURL及びそのウェブページに関するフェッチステータス情報を含む。ウェブインフォメーションディスクファイルへの全てのアクセスは入力バッファを経由してシ

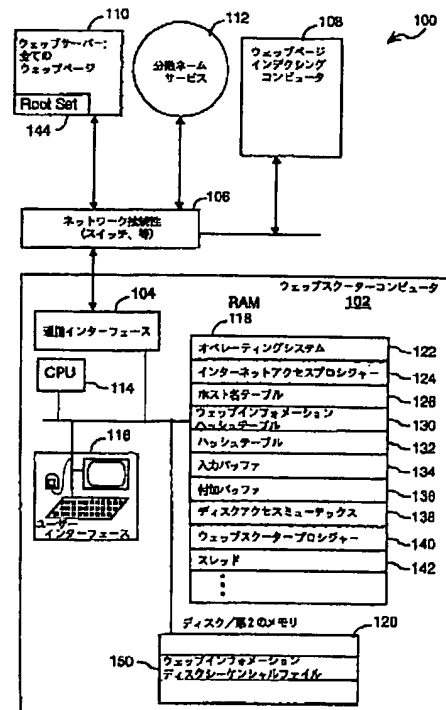


FIG. 1

Best Available Copy

【特許請求の範囲】

1. 各ウェブページは固有のURL（ユニバーサルリソースロケータ）を有し、少なくともいくつかの前記ウェブページは他のウェブページへのURLリンクを含んでいるような、遠隔地に配置されたアクセス可能なコンピュータに記憶されているウェブページを含むデータセットを捜し出すためのシステムであり、

対応するURLに従って、前記の遠隔地に配置されたコンピュータから特定のウェブページをフェッチするための通信インターフェースと、

エントリーの各々が対応するウェブページのURL及びフェッチステータス情報を示すような一セットのエントリーを有するウェブインフォメーションファイルと、

RAM（ランダムアクセスメモリ）に記憶されていて、エントリーの各々が、対応するウェブページの識別値及びフェッチステータス情報を示すようなエントリー一セットを有するウェブインフォメーションテーブルと、

ウェブインフォメーションファイルのエントリーが前記フェッチステータス情報に基づく事前に規定された選定規準を満たすようなウェブページをフェッチするための命令と、受け取られた各々のウェブページ中の各々のURLリンクについて、ウェブインフォメーションテーブル中に対応するエントリーが既に存在するかどうかを決定し、ウェブインフォメーションテーブルに対応するエントリーを有していないURLリンクの各々について、ウェブインフォメーションテーブルに新しいエントリーを加え、ウェブインフォメーションファイルに対応する新しいエントリーを加えるための命令とを含む、前記システムによって実行される、ウェブページをフェッチして解析するためのウェブスクータープロシジャールを実行する手段と

を備えるシステム。

2. 重複する時間期間中に、各々がウェブスクータープロシジャールを実行するような多数のスレッドを含み、スレッドのいくつかがウェブページをフェッチしている間に、ウェブページの他のスレッドは、フェッチされたウェブ

ページを解析しているような手段を含む請求項1に記載のシステム。

3. ミューテックスを含み、スレッドの各々によって実行される前記ウェブスクータープロシジャは、ウェブインフォメーションテーブル及びウェブインフォメーションファイルにアクセスする前にミューテックスを要求して待つための命令を含む請求項2に記載のシステム。

4. 入力バッファ及び付加バッファと、

シーケンシャルに並べられたエントリーのブロックをウェブインフォメーションファイルから入力バッファへ記憶するためのファイルマネージャーと、

入力バッファ中のウェブインフォメーションファイルのエントリーをスキャンして解析し、前記の事前に規定された選定規準を満たす前記ウェブインフォメーションファイルのエントリーを捜し出す前記ウェブスクータープロシジャと、

前記ウェブインフォメーションファイルに加えられるべき全てのエントリーを前記付加バッファに記憶する前記ウェブスクータープロシジャと、付加バッファ中の多数のエントリーをウェブインフォメーションファイルに移すための前記ファイルマネージャーと

を含む請求項3に記載のシステム。

5. 第2のメモリー中のエントリーの各々は第1のメモリー中の対応するエントリーのアドレスを含む請求項1に記載のシステム。

6. 各ウェブページは固有のURL（ユニバーサルリソースロケータ）を有し、少なくともいくつかの前記ウェブページは他のウェブページへのURLリンクを含んでいるような、遠隔地に配置されているがアクセス可能なコンピュータに記憶されているウェブページを含むデータセットを捜し出す方法であり、

各エントリーが、対応するウェブページのURL及びフェッチステータス情報を示すような一セットのエントリーを有するウェブインフォメーションファイルを記憶するステップと、

各エントリーが、対応するウェブページの識別値及びフェッチステータス情報を示すような一セットのエントリーを有するウェブインフォメーション

テーブルをRAM（ランダムアクセスメモリ）に記憶するステップと、

（A）ウェブインフォメーションファイル中のエントリーをシーケンシャルにスキャンして、前記エントリーのどれが事前に規定された選定規準を満たすかを決定し、（B）ウェブインフォメーションファイルのエントリーが前記の事前に規定された選定規準を満たすようなウェブページをフェッチし、（C）受け取られたウェブページの別のウェブページへのURLリンクの各々について、対応するエントリーが既にウェブインフォメーションテーブル中に存在するかどうかを決定し、（D）ウェブインフォメーションテーブル中に対応するエントリーを有していないURLリンクの各々について、ウェブインフォメーションテーブルに新しいエントリーを付加し、ウェブインフォメーションファイルに対応する新しいエントリーを付加することを含む、ウェブページをフェッチして解析するためのウェブスクータープロシジャールを実行するステップと

を備える方法。

7. 重複する時間期間中に多数のスレッドにおいて前記ウェブスクータープロシジャールを実行し、スレッドのいくつかがウェブページをフェッチしている間に、ウェブページの他のスレッドはフェッチされたウェブページを解析するようにすることを含む請求項6に記載の方法。

8. ミューテックスを定義し、

前記スレッドの各々において前記ウェブスクータープロシジャールを実行している間に、ウェブインフォメーションテーブル及びウェブインフォメーションファイルにアクセスする前に、ミューテックスを要求して待つことを含む請求項7に記載の方法。

9. 前記RAMに、「入力バッファ」及び「付加バッファ」を定義し、

シーケンシャルに並べられたエントリーのブロックをウェブインフォメーションファイルから入力バッファへ記憶し、

シーケンシャルにウェブインフォメーションファイルのエントリーをスキャンする前記のステップは、入力バッファのウェブインフォメーションファイルのエントリーをスキャンして、前記ウェブインフォメーションファイル

のエントリーのどれが前記の事前に規定された選定規準を満たすかを決定することを含むステップを備え、

前記ファイルに加えられるべき全てのエントリーを前記付加バッファに記憶し、

付加バッファの多数のエントリーをウェブインフォメーションファイルに移す

ステップを備えている請求項8に記載の方法。

10. ウェブインフォメーションテーブルのエントリーの各々はウェブインフォメーションファイルの対応するエントリーのアドレスを含み、

ウェブインフォメーションテーブルの対応するエントリーのアドレスを読み出して、それから前記アドレスにある前記ウェブインフォメーションファイルの前記の1エントリーを読み出すことによって、前記ウェブインフォメーションファイルの前記エントリーの1つにアクセスすること

を含む請求項6に記載の方法。

11. 各データセットはアドレスによって固有に識別され、少なくともいくつかのデータセットは、コンピュータに記憶された他のデータセットの接続アドレスを1つ以上含むような、ネットワークによって接続されたコンピュータに記憶されているデータセットを捜し出すための装置であり、

識別されたデータセットの要求をコンピュータに送り、前記の要求に応答してデータセットを受け取るための、ネットワークに接続された通信インターフェースと、

各々が対応するデータセットのアドレス及び対応するデータセットのステータス情報を含んでいるようなエントリーの第1の一セットを記憶している第1のメモリと、

各々が対応するデータセットのアドレスの符号化及び対応するデータセットのステータス情報の符号化を含んでいるようなエントリーの第2の一セットを記憶している第2のメモリと、

第1と第2のメモリ及び通信インターフェースに接続され、シーケンシャルに第1の一セットのエントリーを読み出し、事前に規定されたステータスに基

づく選定規準を満たすような対応するエントリーを第1の一セット中に有する識別されたデータセットの要求をつくり、識別されたデータセットを受け取るのに応答して、前記の第1及び第2の一セットに、第2の一セットに対応するエントリーが存在しない受け取られたデータセットの少なくともアドレスの集合の各々に対応する新しいエントリーをつくるスレッド手段と

を備える装置。

12. 第2の一セットのエントリーの各々は第1の一セットの対応するエントリーのアドレスを含み、エントリーの前記の第2の一セットはエントリーの第1の一セットにインデックスを付けるためのものであるような請求項11に記載の装置。

13. スレッド手段のいくつかが前記の要求をつくり、識別されたデータセットを受け取っている間に、他のスレッド手段は前記第1と第2のメモリに新しいエントリーをつくっているような多数の前記スレッドを含む請求項11に記載の装置。

14. ミューテックスを含み、前記スレッド手段の各々は、第1のメモリ及び第2のメモリにアクセスする前にミューテックスを要求して待つロジックを含むような請求項13に記載の装置。

15. 前記第2のメモリに配置された入力バッファ及び付加バッファと、

第1のメモリのシーケンシャルに並べられたエントリーのグループを入力バッファに記憶するマネージャーと、

入力バッファのエントリーをスキャンして解析し、前記の事前に規定されたステータスに基づく選定規準を満たす前記エントリーを捜し出す手段を含む前記スレッド手段の各々と、

前記第1のメモリに加えられるべき全てのエントリーを前記付加バッファに記憶する前記スレッド手段の各々と、

付加バッファの多数のエントリーを第1のメモリに移す手段も有する前記マネージャーと

を含む請求項14に記載の装置。

16. 各データセットはアドレスによって固有に識別され、少なくともいくつかの

前記データセットは、コンピュータに記憶された他のデータセットの接続アドレスを1つ以上含むような、ネットワークによって接続されたコンピュータに記憶されたデータセットを捜し出す方法であり、

(A) 各々が対応するデータセットのアドレス及び対応するデータセットのステータス情報を含んでいるような、エントリーの第1の一セットを第1のメモリに記憶するステップと、

(B) 各々が対応するデータセットのアドレスの符号化及び対応するデータセットのステータス情報の符号化を含んでいるような、エントリーの第2の一セットを第2のメモリに記憶するステップと、

(C) シーケンシャルに第1の一セットのエントリーを読み出すステップと、

(D) 事前に規定されたステータスに基づく選定規準を満たす、第1の一セットの対応するエントリーを有する識別されたデータセットの要求を、ネットワークを経由してコンピュータに伝送するステップと、

(E) 識別されたデータセットを受け取るのに応答して、前記第1及び第2の一セットに、第2の一セットに対応するエントリーが存在しない少なくとも受け取られたデータセットのアドレスの集合の各々に対応する新しいエントリーをつくるステップ

を備える方法。

17. 前記ステップBは、第2の一セットのエントリーの各々に、第1の一セットの対応するエントリーのアドレスを記憶し、エントリーの前記の第2の一セットはエントリーの第1の一セットにインデックスを付けるためのものであるような請求項16に記載の方法。

18. 重複する時間期間中に多数のスレッドにおいてステップC、D、Eを実行し、スレッドのいくつかがデータセットをフェッチしている間に、データセットの他のスレッドは、フェッチされたデータセットを解析しているようにすることを含む請求項16に記載の方法。

19. ミューテックスを定義し、

前記スレッドの各々は、第1及び第2のメモリのエントリーの第1及び第2の一セットにアクセスする前にミューテックスを要求して待つこと

を含むような請求項18に記載の方法。

20. 前記第2のメモリに入力バッファ及び付加バッファを定義し、

シーケンシャルに並べられたエントリーのブロックをエントリーの第1のセットから入力バッファに記憶し、

前記のシーケンシャルに読み出すステップは、入力バッファのエントリーをシーケンシャルに読み出し、前記入力バッファのエントリーのどれが前記の事前に規定されたステータスに基づく選定規準を満たすかを決定するステップを備え、

前記第1のメモリに加えられるべき全てのエントリーを前記付加バッファに記憶し、

付加バッファの多数のエントリーを第1のメモリに移すこと
を含む請求項19に記載の方法。

【発明の詳細な説明】

コンピュータのネットワークからワールドワイドウェブ上のページを捜し出したり、ドキュメントを捜し出したりするためのシステム及び方法

発明の分野

本発明は、一般的に、コンピュータのネットワークからワールドワイドウェブ（WWW）上のページと呼ばれるドキュメントにアクセスしたり、ドキュメントを捜し出したりするためのシステム及び方法に関し、特に、迅速にワールドワイドウェブ上のページを捜し出して解析するためのシステム及び方法に関する。

発明の背景

ここではウェブページと呼ぶウェブドキュメントは、インターネットに接続された多数のサーバーコンピュータ（ここでは以後「サーバー」と呼ぶ。）上に記憶される。ウェブ上の各々のページは別個のURL（ユニバーサルリソースロケータ—universal resource locator）を有する。ウェブサーバー上に記憶された多数のドキュメントはHTML（ハイパーテキストマークアップラングージ—hypertext markup language）と呼ばれる標準のドキュメント記述言語で書かれている。HTMLを使用して、ウェブドキュメントのデザイナーは、ドキュメント中でハイパーテキストリンクもしくはアノテーションをドキュメント中の特定の語又は句と関連付けて、ウェブページの視覚的な外観及び内容を記述する。ハイパーテキストリンクは、その語又は句に関する情報を提供する他のウェブドキュメントもしくは同一のドキュメント中の他の部分のURLを識別する。

インターネットに接続されたウェブクライアント上で動くウェブブラウザー（HTMLドキュメントを表示したり、ウェブサーバーと通信したりするように設計されたコンピュータプログラム）を使用して、ユーザーはWWW上に記

憶されたドキュメントにアクセスする。一般的に、ユーザーが、ウェブブラウザーで表示されるドキュメント内でハイパーテキストリンク（一般的に、強調された語もしくは句としてウェブブラウザーで表示される。）を選ぶことによ

て、ユーザはドキュメントにアクセスする。そして、ウェブブラウザは、要求されたドキュメントのURLによって識別されるウェブサーバーへ、要求されたドキュメントのHTTP（ハイパーテキスト転送プロトコル-hypertext transfer protocol）要求を出す。その要求に応答して、やはりHTTPを使用して、指名されたウェブサーバーは要求されたドキュメントをウェブブラウザに返す。

1995年の末以後、ワールドワイドウェブ（以後、「ウェブ」と呼ぶ。）として知られているインターネット部分のページ数は、先の1年間に数倍にもなり、少なくとも3000万ページに達するようになった。本発明は、ウェブが増大し続ける時に、ウェブ上のページの経路を維持し続けるためのシステムを実施することを意図している。

ウェブ上のページを捜し出すためのシステムは、「ウェブクローラー (Web crawler) 」や「ウェブスパイダー (Web spider) 」や「ウェブロボット (Web robot) 」として様々に知られている。本発明は「ウェブスクーター (Web scooter) 」としてつくられてきた。何故ならば、それは既知のウェブクローラーのどれよりも非常に速いからである。本文では、「ウェブクローラー」、「ウェブスパイダー」、「ウェブスクーター」、「ウェブクローラーコンピュータシステム」、「ウェブスクーターコンピュータシステム」という語を相互に互換性を有する語として使用する。

一般的に、従来技術のウェブクローラーは以下のように動作する。既知のウェブページのルートセットから開始して、全ての既知のウェブページに対する別個のエントリーについて、ディスクファイルがつくられる。更なるウェブページがフェッチされ、他のページへのそれらのリンクが解析される時、まだウェブクローラーに知られていないウェブページを参照するために、ディスクファイルに更なるエントリーがつくられる。エントリーの各々は、他のステータス情報と共に、対応するウェブページが処理されているかどうかを示す。ウェブクローラーはウェブページを次のように処理する。(A) 処理されているページ中の他のウェブページへの全てのリンクを識別し、関連する情報を記憶

して、まだ処理されていない識別された全てのウェブページを、処理されるべきウェブページのリスト、もしくは他の同等のデータ構造に加える。(B)ウェブページをインデクサ、もしくは他のドキュメント処理システムに送る。

一般的に、既に処理されたウェブページに関する情報はディスクファイルに記憶される。何故ならば、ディスクファイルの情報量は、ランダムアクセスメモリ(RAM)に記憶するには大きすぎるからである。例えば、もし平均100バイトの情報がウェブページのエントリーの各々に対して記憶されるならば、3000万ウェブページを表すデータファイルは約3ギガバイトになり、これはRAMに実際に記憶するには大きすぎる。

次に、1ウェブページを処理する時に発生するディスクI/Oについて考える。この説明のために、典型的な1ウェブページは20個の他のウェブページへのリファレンスを有し、ディスク記憶装置は1秒あたり50シークよりも多くの処理はできないと仮定する。ウェブクローラーは、処理されているページ中の20個のページリファレンスの各々を評価して、ウェブクローラーがそれらのページについて既に知っているかどうかを決定しなければならない。これを実施するために、ウェブクローラーはウェブインフォメーションディスクファイルから20個のレコードの検索を試みなければならない。もし、特定ページのリファレンスのレコードが既に存在するならば、そのリファレンスは捨てられる。何故ならば、更なる処理は不要だからである。しかしながら、もし特定ページのレコードが見つからないならば、そのページのアドレスの可能なエイリアスの各々に対してレコードを捜し出す試みがなされなければならない。それによって、標準の1ウェブページを解析するのに必要なディスクレコードの平均シーク数は、1ページあたり約50ディスクシークにまで増大する。

特定ページのリファレンスのディスクファイルのレコードがまだ存在していないならば、参照されたページの新しいレコードがつけられてディスクファイルに加えられる。そして、そのページリファレンスが、処理されるべきページの待ち行列に加えられるか、もしくはそのページがまだフェッチされて処理されていないことを示すのに、そのディスクファイルエントリ自体が使用される。

このように、単に1ウェブページを処理するのに、(存在するレコードを読み出し、新しいレコードを書き込むために) おおよそ20ディスクシークが必要となる。結果として、1秒あたり50ディスクシークの制限を与られているので、1秒あたり約1ウェブページしか処理されない。

加えて、ネットワークアクセス待ち時間の問題がある。ウェブサーバー及びウェブサーバーとウェブクローラーコンピュータ上の両方で使用される特定のハードウェアとソフトウェアの位置によって、ウェブページを検索する時間は大きく変わるけれども、平均的に、ウェブページを検索するのには約3秒かかる。このように、ネットワーク待ち時間もやはり、従来技術のウェブクローラーによって処理されるウェブページ数を1秒あたり約0.33ウェブページに制限する恐れがある。ディスク「シーク」制限、ネットワーク待ち時間、及び他の遅延要因のために、代表的な従来技術のウェブクローラーは1日あたり約30,000ウェブページよりも多くのページを処理することができない。

ウェブページがウェブに追加される速度、及びウェブページが削除されたり、改訂されたりする速度の理由から、1日あたり30,000ウェブページの処理では、ウェブ上の全てのウェブページの真に最新のディレクトリもしくはインデックスを維持するには不十分である。理想的には、ウェブクローラーは1日あたり少なくとも250万ウェブページを訪れる(すなわち、フェッチ及び解析する)ことができないなければならない。

従って、非常に高速の性能を持つウェブクローラが必要となる。本発明の目的は、1日あたり数百万のウェブページを処理することができるよう改良されたウェブクローラーを提供することである。本発明の関連する目的は、主に、ウェブクローラーのCPUの処理速度によってのみ、ウェブクローラーの動作速度が制限されるようにするために、前記のディスク「シーク」制限及びネットワーク待ち時間制限を解決するような改良されたウェブクローラーを提供することである。更に、本発明の別の関連する目的は、平均して、1秒あたり少なくとも30ウェブページ、より好ましくは1秒あたり少なくとも100ウェブページをフェッチ及び解析することができるウェブクローラーシステムを提

供することである。

発明の概要

本発明の本質は、請求項1に記述されているようなウェブページを捜し出すためのシステム及び請求項6に記述されているようなウェブページを捜し出すための方法に存在する。

以後に、迅速に、ワールドワイドウェブ上のウェブページのディレクトリを捜し出して作成するためのシステム及び方法を説明する。ウェブクロウラーシステムは、ランダムアクセスメモリ(RAM)に記憶されたハッシュテーブル及び代表的なディスク記憶装置である第2のメモリに記憶されたシーケンシャルファイル(ここでは、「シーケンシャルディスクファイル」もしくは「ウェブインフォメーションディスクファイル」と呼ぶ。)を含む。システムにとって既知である全てのウェブページに対して、ウェブクロウラーシステムは、ハッシュテーブルにより小さなエントリーを記憶するのに加えて、シーケンシャルディスクファイルにエントリーを記憶する。ハッシュテーブルのエントリーは、識別値、対応するウェブページが上手くフェッチされた時だけ真にセットされるフェッチフラグ、対応するエントリーがシーケンシャルディスクファイルのどこに記憶されているかを示すファイル位置インジケータを含む。シーケンシャルディスクファイルエントリーの各々は、対応するウェブページのURLとそのウェブページに関するフェッチステータス情報を含む。

ウェブインフォメーションディスクファイルへの全てのアクセスは、単一のI/Oオペレーションとして、シーケンシャルディスクファイルから多数のエントリーを入力バッファに移すといったように、入力バッファを経由して、シーケンシャルに行われる。従って、シーケンシャルディスクファイルは入力バッファからアクセスされる。同様に、シーケンシャルファイルに加えられる全ての新しいエントリーは付加バッファに記憶され、付加バッファが一杯になった時はいつでも、付加バッファの内容はシーケンシャルディスクファイルの最後に加えられる。このようにして、ウェブインフォメーションディスクファイルへのランダムア

クセスは排除され、ディスクアクセス制限によって引き起こされる待ち時間は最小化される。

ウェブページを捜し出して、そのページを処理するためのプロシジャは、シーケンシャルにシーケンシャルファイルの全てのエントリーを見直し、設定された選定規準を満たす次のエントリーを選ぶことを含む。処理する次のファイルエントリーを選ぶ時、ハッシュテーブルは、最新のエントリー候補の全ての既知のエイリアスと照合されて、エイリアスでそのウェブページが既にフェッチされているかどうかを決定する。もしエイリアス下でそのウェブページは既にフェッチされているならば、そのシーケンシャルファイルのエントリーのエラータイプフィールドは「非一選定エイリアス」として記録され、その候補エントリーは選ばれない。

一度、次のウェブページのリファレンスエントリーが選ばれると、ウェブクローラーシステムは対応するウェブページへのフェッチを試みる。もし、フェッチが不成功ならば、そのウェブページのシーケンシャルファイルエントリーのフェッチステータス情報は、ウェブクローラーに返されたエラーリターンコードに従ってフェッチ失敗として記録される。もし、フェッチが成功ならば、そのウェブページの（入力バッファ中の）シーケンシャルディスクファイルのエントリーの類似のフェッチフラグと同様に、そのウェブページのハッシュテーブルエントリーのフェッチフラグがセットされる。加えて、フェッチされたウェブページ中のURLリンクの各々が解析される。もしそのリンクによって参照されるURLもしくはそのURLの規定されたエイリアスのいずれかのエントリーが既にハッシュテーブルにあるならば、そのURLリンクの更なる処理は必要ではない。もしこのようなエントリーがハッシュテーブル中に見つからないならば、そのURLは、まだウェブページのウェブクローラのデータベースに含まれていない「新しい」ウェブページを表しており、従って、新しいウェブページのエントリーがシーケンシャルディスクファイルに加えられる（すなわち、そのエントリーは付加バッファのディスクファイルの一部に加えられる。）その新しいディスクファイルのエントリーは処理されているリンクによって参照されるURLを含み、「未フェッチ」と登録される。加えて、対応する新しい

エ

ントリーがハッシュテーブルに加えられ、そしてそのエントリーのフェッチフラグはクリアされて、対応するウェブページはまだフェッチされていないことを示すようにする。フェッチされたページ中の全てのURLリンクの処理に加えて、更なる処理のために、ウェブクローラーはフェッチされたページをインデクサーに送る。

図面の簡単な説明

添付図を参照して例示された以下の好ましい実施態様の説明によって、本発明のより詳細な理解が得られるであろう。

- ・図1は本発明の好ましい実施態様に従うウェブクローラーシステムの好ましい実施態様のブロック図である。

- ・図2は本発明の好ましい実施態様で使用されるハッシュテーブルの仕組みのブロック図である。

- ・図3は本発明の好ましい実施態様で使用されるシーケンシャルなウェブインフォメーションディスクファイル及び関連するデータ構造のブロック図である。

- ・図4は本発明の好ましい実施態様で使用されるウェブクローラープロシージャのフローチャートである。

好ましい実施態様の説明

図1について、ウェブスクーターコンピュータシステム102を含む分散コンピュータシステム100が示されている。通信インターフェース104及び一セットのインターネットや他のネットワークへの接続106によって、ウェブスクーターはインターネットやウェブページインデクシングコンピュータ (Web page indexing computer) 108に接続される。ある実施態様では、ウェブページインデクシングコンピュータ108は、ローカルもしくはワイドエリアネットワーク接続を使用せずに、専用通信チャンネルを通して、ウェブスク

ーター102に直接接続される。ウェブスクーター102が接続されるインタ

ーネット部分は、(A) ウェブページを記憶するウェブサーバー110及び
(B) ここでは総体的に参照番号112で参照されている、分散ネームサービス
(DNS)として知られているサービスに協力するサーバーである。本文では、
DNS112は、全てのインターネットのホスト名に対して規定された全てのエイ
リアスセットを全ての要求者に提供し、そしてインターネットのホスト名及
びそれらのエイリアスは、全てのURLの先頭部分を形成すると仮定する。

好ましい実施態様では、ウェブスクーター102はデジタルイクイップメン
ト社製のアルファワークステーションコンピューターであるが、実際は、あらゆる
タイプのコンピューターをウェブスクーターコンピュータとして使用すること
ができる。好ましい実施態様では、ウェブスクーター102はCPU114
、前記の通信インターフェース104、ユーザーインターフェース116、ラン
ダムアクセスメモリ(RAM)118、ディスクメモリ(disk)120を含
む。好ましい実施態様では、通信インターフェース104は非常に高性能の通信
インターフェースであり、1秒あたり少なくとも30ウェブページの平均フェ
ッチスループットで、1000以上の重複する通信要求を取り扱うことができる
。

好ましい実施態様では、ウェブスクーターのRAMは1ギガバイトのランダ
ムアクセスメモリを有し、以下のものを記憶する。

- ・マルチタスクオペレーティングシステム122。
- ・DNS112からエイリアス情報をフェッチするためであるのに加えて、ウ
ェブページをフェッチするためでもあるインターネット通信マネージャープロ
グラム124。
- ・ホスト名に対して規定されたエイリアスを表す情報を記憶するホスト名テー
ブル126。
- ・ウェブインフォメーションハッシュテーブル130。
- ・ハッシュテーブルマネージャープロシジャー132。
- ・入力バッファ134及び付加バッファ136。
- ・ハッシュテーブル130、入力バッファ134、付加バッファ136へのア
クセスを制御するためのミューテックス(mutex)138。

- ・ ウェブスクータープロシジャー140。
- ・ T1スレッドの実行を決定するためのスレッドデータストラクチャー142。

ここで、T1の値はウェブスクーターコンピューターシステム102のオペレーターが選ぶことができる整数である。(例えば、好ましい実施態様では、T1は1000の値にセットされる。)

より詳細を以下に説明するように、ディスク記憶装置120は、入力バッファ134及び付加バッファ136を経由して、シーケンシャルにアクセスされるウェブインフォメーションディスクファイル150を記憶する。

ホスト名テーブル126は、特に、DNS112にとって既知である各ホスト名の全てのエイリアスを表す情報を記憶する。エイリアスは、ウェブスクータープロシジャー140によって特定のウェブページのURLのホスト名部分の代わりにされる効果的なURLの先頭部分の一セットであり、特定のウェブページのエイリアスのURLの一セットを成す。

次に、上記データ構造及びプロシジャーの使用及び動作を図1-図4及び表1-表2を参照して説明する。表1-表2は共にウェブスクータープロシジャーの疑似コード表現を含む。ここで使用される疑似コードは、この説明の目的のためだけにつくられているが、その疑似コードは一般的なコンピュータ言語の規約を使用しており、当業者である全てのコンピュータプログラマーが容易に理解可能であるように設計されている。

ウェブインフォメーションハッシュテーブル

図2に関して、ウェブインフォメーションハッシュテーブル130は、フェッチされて解析されたウェブページのURLリンクによって参照される各ウェブページに加えて、ウェブスクーターシステムによってフェッチされて解析された各ウェブページに対する別個のエントリー160を含む。このようなエントリーの各々は、以下を含む。

- ・ 対応するウェブページに固有な識別値162。
- ・ 対応するウェブページがウェブスクーターによってフェッチされて解析

されたかどうかを示す1ビットの「フェッチフラグ」164。

・ウェブインフォメーションディスクファイル150の対応するエントリーの位置を示すファイル位置値166。

好ましい実施態様では、識別値の各々は63ビットの長さであり、ファイル位置値は各々32ビットの長さである。結果として、好ましい実施態様において、ハッシュテーブルエントリーの各々は丁度12バイトを占める。ハッシュテーブルエントリーのサイズそのものは重要でないが、ハッシュテーブルエントリー160の各々は対応するディスクファイルエントリーよりもかなり小さい（例えば、平均して少なくとも75%小さい。）ということは重要である。

ハッシュテーブルマネージャー132は、その「インターフェース」170を経由して、ウェブスクータープロシジャー140から以下の2種類のプロシジャーコールを受け取る。

・第1の要求はハッシュテーブルマネージャー132に特定のURLのエントリーが存在するかどうかを問い合わせ、そして、もし存在するならば、その記録のフェッチフラグが、対応するウェブページが先にフェッチされて解析されたことを示すかどうかを問い合わせる。

・第2の要求は、特定のURL及び特定のディスクファイル位置の新しいエントリーをハッシュテーブル130に記憶するようにハッシュテーブルマネージャーに要求する。

ハッシュテーブルマネージャー132は識別ハッシュ関数172を使用して、そこに現れる全てのURLの63ビットの識別値を計算する。識別関数172は、確実に全ての固有のURLが同様に固有の識別値に変換されるように設計されている。識別関数は全ての固有のウェブページのURLの圧縮されたコードをつくる。通常の当業者であれば、適切な識別関数の設計を理解している。約 2^{25} から 2^{26} のウェブページがあると、識別値は 2^{63} の別個の値を持つことが可能であるということを注記する。

ハッシュテーブルが既に固有のURLのエントリーを有するかどうかを、ウェブスクータープロシジャー140がハッシュテーブルマネージャー132に問い合わせる時、ハッシュテーブルマネージャーは、(A)前記の識別ハッシュ関

数172を使用して、固有のURLの識別値をつくる。(B) ハッシュテーブル130のどこにその識別値を有するエントリーを記憶するかを決定するハッシュテーブル位置関数174にその値を送る。(C) 実際に、このようなエントリーがハッシュテーブルに記憶されているかどうかを決定する。(D) もしマッチするエントリーが見つからないならば、失敗値(例えば、-1)を返す。(E) もしハッシュテーブルにそのエントリーが見つかったならば、成功値(例えば、0)及びそのエントリーのフェッチフラグ値とディスク位置値を返す。

好ましい実施態様では、識別値の所定数の低位ビットに基づいて、ハッシュテーブル位置関数174はハッシュテーブルエントリーの位置を決定し、同一の低位ビットを持つ全ての識別値のエントリーのブロックのチェーンに続く。ハッシュテーブル130中の、与えられた値の低位ビットのエントリー160は、1ブロックあたりB1エントリーのブロックに配置される。ここで、B1は調整可能なパラメーターである。好ましい実施態様で使用する上記の方法は、ハッシュテーブル130に高密度な方法でデータを記憶するという利点がある。当業者は理解しているように、多くの他のハッシュテーブル位置関数を使用することができる。

ウェブスクータープロシジャー140が、ハッシュテーブルマネージャー132に特定のURL及び特定のディスクファイル位置の新しいハッシュテーブルのエントリーを記憶することを要求する時、ハッシュテーブルマネージャーは、(A) 前記の識別ハッシュ関数172を使用して、特定のURLの識別値をつくる。(B) ハッシュテーブル130のどこに識別値を有するエントリーを記憶しなければならないかを決定するハッシュテーブル位置関数174にその値を送る。(C) ハッシュテーブルの所定の位置に、対応するウェブページがまだフェッチされていないことを示すフェッチフラグ値、識別値、特定のディスクファイル位置と共に新しいエントリー160を記憶する。

ウェブインフォメーションディスクファイル及びバッファ

図3及び表2に関して、入力バッファ134及び付加バッファ136は、どち

らのバッファもRAMに配置されており、これらのバッファの使用によって、デ

ディスクアクセスの動作は最小化される。入力バッファ及び付加バッファの管理は、ディスクファイルマネージャーとしても知られているバックグラウンドのシーケンシャルディスクファイル及びバッファハンドラプロシジャによって実施される。

好ましい実施態様では、入力バッファ及び付加バッファはサイズが各々50から100メガバイトである。入力バッファ134は、ウェブインフォメーションディスクファイル150の、シーケンシャルに並べられた連続部分を記憶するのに使用される。ウェブスクータープロシジャは、入力バッファ134、付加バッファ136、ディスクファイル150の使用の調整を要求される多数の他のブックキーピングポインター (bookkeeping pointer) に加えて、入力バッファの処理されるべき次のエントリーへのポインター176、及びウェブインフォメーションディスクファイル150の入力バッファ134に転送されるべき次のエントリー180へのポインター178を維持する。

単一のI/O動作として、多数のエントリーがシーケンシャルディスクファイルから入力バッファへ移されるといったように、ウェブインフォメーションディスクファイル150への全てのアクセスは、入力バッファ134を経由して、シーケンシャルに行われる。従って、シーケンシャルディスクファイル150は入力バッファからアクセスされる。同様に、シーケンシャルファイルに加えられる全ての新しいエントリーは付加バッファ136に記憶され、付加バッファが一杯になった時はいつでも、付加バッファの内容がシーケンシャルファイルの最後に加えられる。このようにして、ウェブインフォメーションディスクファイルへのランダムアクセスは減少され、ディスクアクセス制限によって引き起こされる待ち時間は最小化される。

ウェブスクーターによって入力バッファ134の全てのエントリーがスキャンされる度に、入力バッファのエントリーへの全ての更新がウェブインフォメーションディスクファイル150に再度記憶され、付加バッファ136の全てのエントリーがディスクファイル150の最後に加えられる。加えて、付加バッファ136はクリアされ、ディスクファイルのエントリーの次の一セットが、(ポ

インター178によって示される) 入力バッファ134にコピーされるべきエントリーの最後の一セットの直後から、入力バッファ134にコピーされる。ウェブスクータープロシジャーによってディスクファイルの最後のエントリーがスキャンされると、スキャンはディスクファイル150の先頭に戻る。

付加バッファ136が新しいエントリーで一杯になった時はいつでも、その中身はディスクファイル150の最後に加えられ、そして、付加バッファはクリアされて新しいエントリーを受け取る。

ウェブインフォメーションディスクファイル150のエントリー180の各々は以下を記憶する。

- ・エントリーによって参照されるウェブページのURLを記憶する可変長のURLフィールド182。
- ・ウェブスクーターによって、対応するウェブページがフェッチされて解析されたかどうかを示すフェッチフラグ184。
- ・参照されたウェブページがフェッチされ、解析され、そしてインデックスを付けられた日付及び時間を示すタイムスタンプ186。
- ・ウェブページのサイズを示すサイズ値188。
- ・もし何か、エントリーが重複している(すなわち、エイリアスのURLの)エントリーで、無視されるべきであるといったような場合に、又は、参照されるウェブページにフェッチする最後の試みが行われた時に発生したエラータイプを示すエラータイプ値190。
- ・ここでは取り上げない他のフェッチステータスパラメーター192。

URLフィールド182は可変長であるので、ウェブインフォメーションディスクファイル150のレコード180もまた可変長である。

ウェブスクータープロシジャー

図1-図4及び表1の疑似コードに関して、好ましい実施態様におけるウェブスクータープロシジャー140は以下の通り動作する。ウェブスクータープロシジャーが実行を開始する時、そのプロシジャーはシステムのデータ構造を初

期化する(200)。

- ・既に存在しているウェブインフォメーションディスクファイル150をスキャンし、シーケンシャルファイルの全てのエントリーに対するエントリーについてハッシュテーブル130を初期化する。

- ・シーケンシャルディスクのエントリーの第1のバッチをディスクファイル150から入力バッファ134にコピーする。

- ・空の付加バッファ136を新しいシーケンシャルファイルのエントリー用に定義する。

- ・入力バッファ134、付加バッファ136、ハッシュテーブル130へのアクセスを制御するためのミューテックス138を定義する。

それから、ウェブスクーターイニシャライザーはT1個のスレッドを開始し（例えば、好ましい実施態様では1000個のスレッドが開始される。）、スレッドの各々は同一のスクータープロシジャールを実施する。

ウェブスクーターイニシャライザールプロシジャールの実施の前に既に存在しているウェブインフォメーションディスクファイル150のエントリーセツトは既知のウェブページの「ルツセツト」144と呼ばれる。「アクセス可能」なウェブページのセツトは、ルツセツト中のURLリンクによって参照される全てのウェブページ及び他のアクセス可能なウェブページ中のURLリンクによって参照される全てのウェブページから成る。このように、いくつかのウェブページはウェブスクーター102にとってアクセス不可であるようにすることが可能である。何故ならば、ルツセツトと「アクセス不可な」ウェブページとの間にはURLリンクがないからである。

様々なチャネルによって、このようなウェブページに関する情報が使用可能になると、更なるエントリーの「マニュアル」挿入もしくは更なるエントリーを含むための他の仕組みによって、ウェブインフォメーションディスクファイル150を拡張することができ（それによって、ルツセツト144を拡張する。）、以前にアクセス不可であったウェブページをアクセス可能にする。

以下は、全ての同時に実行されるスレッドによって実行されるウェブスクータープロシジャールの説明である。プロシジャールの第1のステップはミューテック

ス(202)を要求して待つことである。ミューテックスの所有権が要求され、2つのスレッドが同一のディスクファイルのエントリーを処理しないように、そして2つのスレッドが同時にハッシュテーブル、入力バッファ、不可バッファもしくはディスクファイルへの情報の書き込みを試みないようにする。ハッシュテーブル130、入力バッファ134、付加バッファ136、ディスクファイル150はここでは総合的に「保護されたデータ構造」と呼ばれる。なぜならば、ミューテックスの使用によって、それらは総合的に保護されているからである。一度スレッドがミューテックスを所有すると、そのスレッドが規定された選定規準(204)を満たすエントリーを捜し出し、そのエントリーを選ぶまで、そのスレッドは入力バッファ中のディスクファイルのエントリーを(ポインター176によって示される)まだスキャンされていない次のエントリーからスキャンする(204)。

例えば、デフォルトの選定規準は次の通りである。「エントリーによって、一度もフェッチされていない、もしくは最後にフェッチされて解析されたのがH1時間よりも以前であると示されているようなウェブページを参照する全てのエントリー。ここで、H1はオペレーターが選定可能な値である。ただし、エントリーは重複しているエントリーであることをエラータイプフィールドが示している(すなわち、以下に説明されるように「非-選定エイリアス」である。)エントリーを除く。」もしH1が168にセットされるならば、最後にフェッチされて解析されたのが1週間よりも前であるようなウェブページを参照する全てのエントリーが選定規準を満たす。ウェブページの大きさが考慮されるような選定規準の別の例は次の通りである。「一度もフェッチされていないウェブページ、もしくは最後にフェッチされて解析されたのがH1時間よりも前であり、大きさがS1よりも大きいようなウェブページ、もしくは最後にフェッチされて解析されたのがH2時間よりも前であり、大きさがS1以下であるようなウェブページを表しているエントリー。ただし、エントリーが「非-選定エイリアス」であることをエラータイプフィールドが示しているエントリーを除く。ここでS1、H1、H2はオペレーターが選定可能な値である。」

処理すべき次のエントリーを選定する時、ハッシュテーブルを検索して現在の

エントリー候補の全ての既知のエイリアスを見つけ、エイリアスでそのウェブページが既にフェッチされたかどうかを決定する。特に、もしエントリーが、規定された選定規準を満たすならば、ホスト名テーブル126の情報を使用して、そのエントリーのURLの全ての既知のエイリアスがつくられ、それからハッシュテーブル130が検索され、参照されたウェブページがそのエイリアスのURL下でフェッチされたことを示すフェッチフラグを有するエイリアスのURLのいずれかのエントリーを、そのハッシュテーブルが記憶しているかどうかを調べる。もし入力バッファ中の現在のエントリー候補によって参照されるウェブページが、エイリアスのURL下で既にフェッチされていると判断されるならば、その入力バッファのエントリーのエラータイプフィールド190は変更され、このエントリーは「非選定エイリアス」であるとするようにする。このようにして、今回及び以後、エントリーが更なる処理のために選定されるのを防ぐ。

一度、ウェブページのリファレンスエントリーが選定されると、ミューテックスは解放され、他のスレッドが保護されたデータ構造にアクセス可能となる(206)。それから、ウェブスクータープロシジャは対応するウェブページをフェッチするのを試みる(208)。フェッチが成功したか、もしくは失敗した後、再度、そのプロシジャはミューテックスを要求して待ち(210)、再度、そのプロシジャが保護されたデータ構造を使用できるようにする。

もしフェッチが不成功ならば(212-N)、そのウェブページのシーケンシャルファイルのエントリー中のフェッチステータス情報を、ウェブクロラーへ返されたエラーリターンコードに従ってフェッチ失敗として記録する(214)。もしフェッチが成功ならば(212-Y)、(入力バッファの)シーケンシャルディスクファイルのエントリー180中のそのウェブページのフェッチフラグ184のように、ハッシュテーブルのエントリー160中のそのウェブページのフェッチフラグ164がセットされる。加えて、フェッチされたウェブページのURLリンクの各々が解析される(216)。

フェッチされたウェブページが解析された、もしくはフェッチ失敗が入力バッファのエントリーに記録された後、ミューテックスは解放され、他のスレッド

が保護されたデータ構造にアクセスできるようにする(218)。

次に、フェッチされたウェブページのURLリンクを解析するためのプロシジャを図4Bを参照して説明する。ウェブページは、インデクシングシステム108によってインデックスを付けるための適切な情報を保有していない画像ファイルのようなドキュメントへのURLリンクを保有することができるということをここで注記する。しばしば、これらの参照されるドキュメントは、それらを参照するウェブページの構成要素として使用される。本文では、画像ファイルや他のインデックス付け不可ファイルのような構成要素のファイルへのURLリンクは、「他のウェブページへのURLリンク」とはしない。インデックス付け不可ファイルへのこれらのURLリンクは、ウェブスクータープロシジャによって無視される。

一度、他のウェブページに接続する全てのURLを処理してしまうと(230)、インデックスを付けるためのインデクサーにフェッチされたウェブページを送り(232)、ウェブスクーターによる、フェッチされたウェブページの処理を完了する。そうでない場合には、ウェブページへの次のURLリンクが選定される(234)。もし選定されたリンクに関連するURLのハッシュテーブルのエントリが既に存在するならば、そのリンクの更なる処理を要求せず、もし解析されているウェブページにどれか未処理のURLリンクが残っているならば、次のURLリンクを選定する(234)。

もし選定されたリンクに関連するURLのハッシュテーブルのエントリがまだ存在しないならば、ホスト名テーブル126の情報を使用して、そのエントリのURLの全ての既知のエイリアスをつくる。それから、ハッシュテーブル130を検索し、そのテーブルが、そのエイリアスのURLのいずれかのエントリを記憶しているかどうかを調べる(238)。もしハッシュテーブル中にそのエイリアスのURLのいずれかのエントリが存在するならば、そのリンクの更なる処理を要求せず、そしてもし解析されたウェブページにどれか未処理のURLリンクが残っているならば、次のURLリンクを選定する(234)。

もしハッシュテーブル中に選定されたリンクのURLもしくはそのエイリアスのいずれかのエントリが見つからないならば、そのURLは、まだウェブペ

ージのウェブクロウラーのデータベースに含まれていない「新しい」ウェブページを表し、従って新しいウェブページのエントリーが、付加バッファ中のディスクファイルの一部に加えられる(240)。その新しいディスクファイルのエントリーは処理されたリンクによって参照されるURLを含み、「未フェッチ」と記録される。加えて、対応する新しいエントリーがハッシュテーブルに加えられ、そしてそのエントリーのフェッチフラグはクリアされて、対応するウェブページはまだフェッチされていないことを示すようにする(240)。それから、もしウェブページ中にどれか未処理のURLリンクが存在するならば、ウェブページの処理はウェブページの次の未処理のURLリンクについて継続する。

目的及び動作が本文の範囲外であるようなプロシジャによって、ウェブインフォメーションディスクファイル150へのインデックスとして、ウェブインフォメーションハッシュテーブル130は使用される。何故ならば、ハッシュテーブル130は、既知のウェブページの各々のディスクファイル位置の値を保有するからである。いいかえると、ウェブインフォメーションハッシュテーブル中の対応するエントリーのディスクファイルアドレスを最初に読み出し、それからそのアドレスにあるウェブインフォメーションディスクファイルのエントリーを読み出すことによって、ウェブインフォメーションディスクファイル中のエントリーはアクセスされる。

他の実施態様

好ましい実施態様のハッシュテーブル構造130の代わりに、バランストリー(balanced tree)、スキップリスト(skip list)といったような、ウェブインフォメーションハッシュテーブル130の同一の属性を有する全てのデータ構造を使用することが可能である。

解法として、本発明は3つの基本の仕組みを使用して、従来技術のウェブクロウラーの速度制限を克服している。

第1に、どのウェブページリンクがまだウェブクロウラーに知られていな

い新しいウェブページを表しているかを決定するのに十分な情報を含むウェッ

ブページディレクトリテーブルが、RAM中に記憶され、ディスクファイルにアクセスする必要なく、受け取られたウェブページを解析できるようにしている。

第2に、より完全なウェブページディレクトリはシーケンシャルな順番にだけアクセスされ、ディスクアクセスがウェブクローラーの実施速度に重要な影響を持たない程度まで、実施されるディスクアクセスの数を減少するような大きな入力及び付加バッファによって、それらのアクセスを実施する。

第3に、ウェブスクータープロシジャを実行するための多数の同時にアクティブなスレッドを使用し、そしてウェブサーバーへの、同様の数の同時の通信チャンネルを操作可能な通信インターフェースを備えることによって、本発明はネットワークアクセス待ち時間によって引き起こされる遅延を避ける。

特に、多数のスレッドがウェブページフェッチ要求に対する応答を待っている間に、他のスレッドは受け取ったウェブページを解析している。同一のウェブスクータープロシジャを実施する多数のスレッドを使用することによって、受け取られたウェブページを処理できるようになるためにミューテックスを待っている受け取られたウェブページに関するスレッドの待ち行列が、平均的に存在するようである。また、ウェブページのフェッチは、時間的にずれて行われる傾向にある。結果として、ウェブスクーターは殆どウェブページを受け取るために待っていたり、他にやる仕事がないという状態にはならない。マルチプロセッサのワークステーションを使用し、そしてウェブスクータープロシジャを同時に実行するスレッドの数を更に増大することによって、ウェブスクーターのスループットを更に増大することが可能である。

いくつかの特定の実施態様を参照して本発明を説明したが、この説明は本発明の例であり、本発明を限定するものとして解釈されてはならない。ここで提示され、請求された本発明の範囲から逸脱することなく、さまざまな改修が可能である。

表 1

ウェブスクータープロセスの疑似コード表現

プロセス：ウェブスクーター

```
{
  /*初期化ステップ*/
  既に存在しているウェブインフォメーションディスクファイル中をスキャンし、
    シーケンシャルファイル中の全てのエントリーに対するエントリーについてハ
    ッシュテーブルを初期化する。
  シーケンシャルディスクのエントリーの第1のバッチをRAMの入力バッファ中
    に読み出す。
  新しいシーケンシャルファイルのエントリー用の空の付加バッファを定義する。
  入力バッファ、付加バッファ、ハッシュテーブルへのアクセスを制御するための
    ミューテックスを定義する。
  1000 スレッドを開始する。各々のスレッドは同一のスクータープロセス
    を実行する。
}
```

プロセス：スクーター

```
{
  ドゥ フォーエバー (Do Forever) :
    {
      ミューテックスを要求して待つ。
      設定されたURL選定規準に従って、処理すべき新しいURLが選定さ
        れるまで、(入力バッファの) シーケンシャルファイルを読み出す。処
        理する次のURLを選定する時、ハッシュテーブルを調べてURLの全
        ての既知のエイリアスを検索し、そのウェブページがエイリアスで既
        にフェッチされているかどうかを決定する。そして、もしそのウェブ
```

ページがエイリアスでフェッチされているならば、シーケンシャルファイルのエントリーのエラータイプフィールドに「非-選定エイリアス」として記録する。

／*選定規準の例：URLは一度もフェッチされていないか、もしくは最後にフェッチされたのはH1時間よりも前であり、かつ非-選定エイリアスではない。*/

ミューテックスを解放する。

選定されたウェブページをフェッチする。

ミューテックスを要求して待つ。

もしフェッチが成功ならば、

{

ハッシュテーブルのエントリー及び入力バッファのシーケンシャルファイルのエントリーにおいてページをフェッチ済として記録する。

／*フェッチされたページを解析する。*/

ページのURLリンクの各々に対して

{

もしURLもしくはいずれかの規定されたエイリアスが既にハッシュテーブルに存在するならば、

{何も行わない。}

他の場合には、

{

／*そのURLは、まだデータベースに保有されていない「新しい」ウェブページを表す。*/

「未フェッチ」と記録されたエントリーと共に、対応するウェブページの新しいエントリーを付加バッファに加える

。

「未フェッチ」と記録されたエントリーと共に、エントリーをハッシュテーブルに加える。

}

}

処理するために、フェッチされたページをインデクサーに送る。

}

他の場合には、

{

現在処理されている入力バッファ中のエントリーに、受け取ったリターンコードに基づく適切な「フェッチ失敗」エラーインジケータを記録する。

}

ミューテックスを解放する。

} /*ドゥ フォーエバーループの終了*/

}

表 2

バックグラウンドシーケンシャルファイルバッファハンドラーの
疑似コード表現

プロシジャー：バックグラウンドシーケンシャルファイルバッファハンドラー（
a/k/a ディスクファイルマネージャー）

{

「読み出しシーケンシャルファイル」命令が入力バッファをオーバーフローさせる時はいつでも、

{

入力バッファをシーケンシャルディスクファイルに再コピーする。

次のエントリーセットを入力バッファに読み出す。

付加バッファの内容をシーケンシャルディスクファイルの最後に付加する。

付加バッファをクリアして、新しいエントリーに備える。

}

「シーケンシャルファイルへのエントリーの付加」が付加バッファのオーバーフローを引き起こす時はいつでも、

{

付加バッファの内容をシーケンシャルディスクファイルの最後に付加する。

付加バッファをクリアして、新しいエントリーに備える。

処理が終了していない新しいエントリーを付加バッファの先頭に加える。

}

}

【図1】

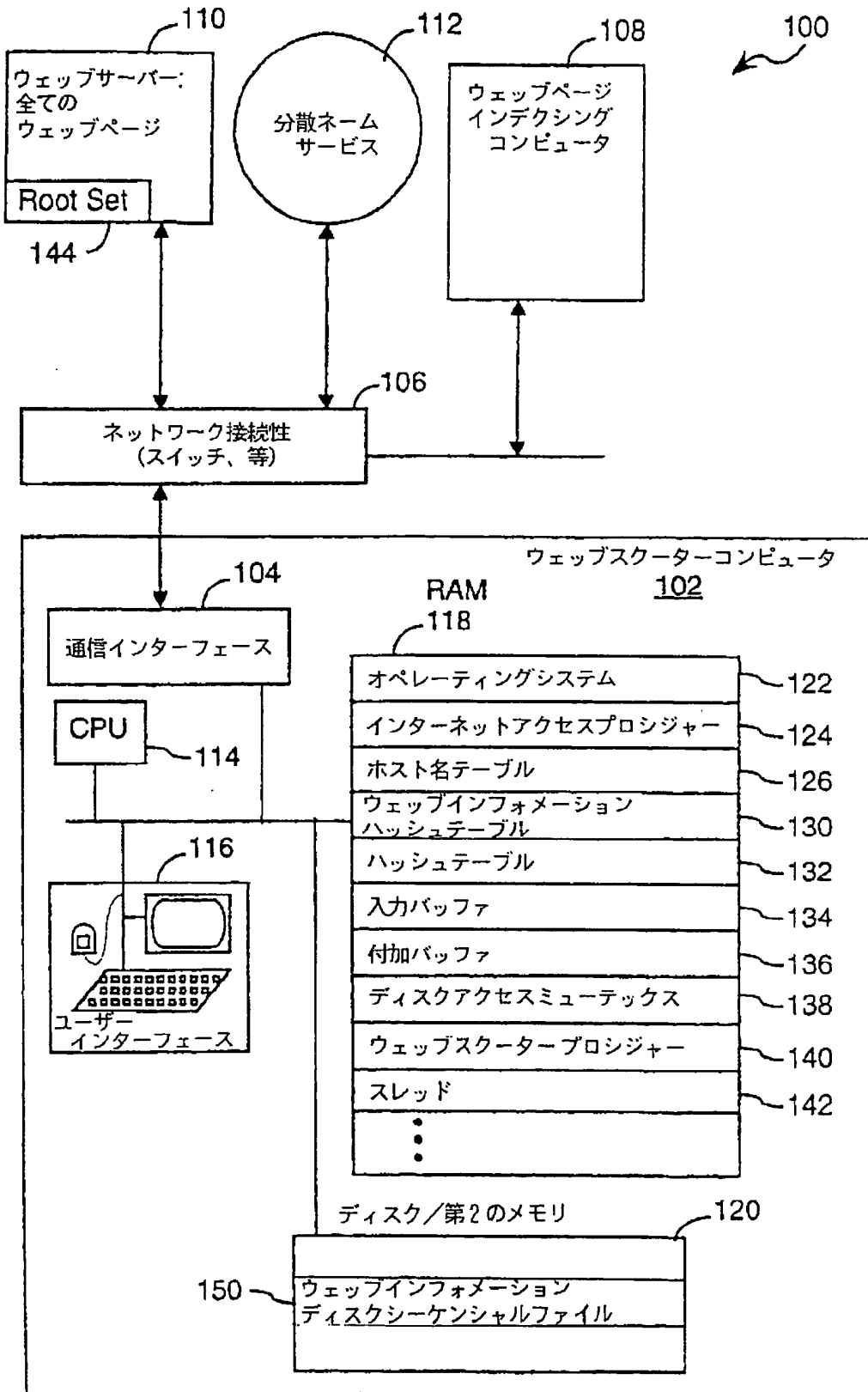


FIG. 1

【図2】

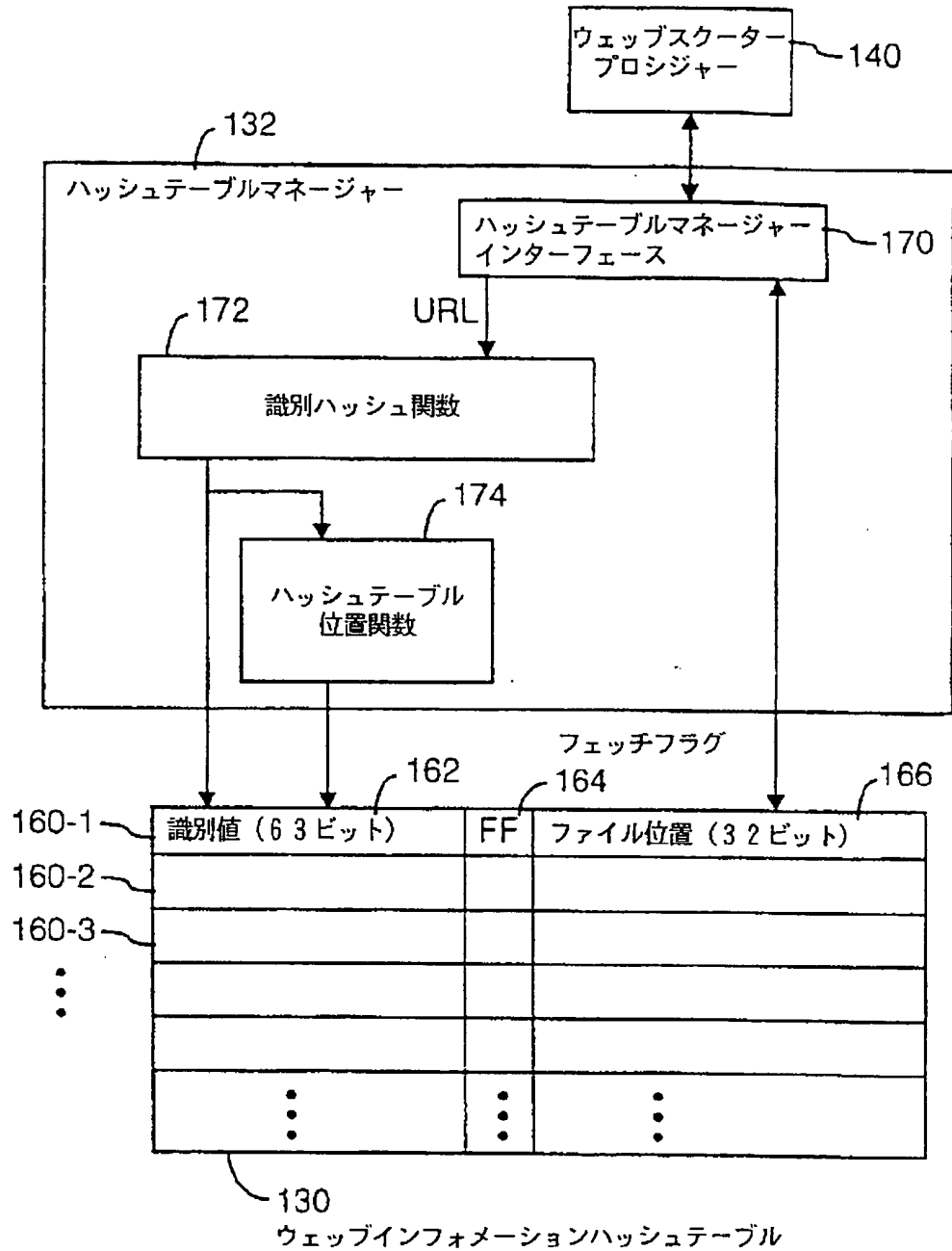


FIG. 2

【図3】

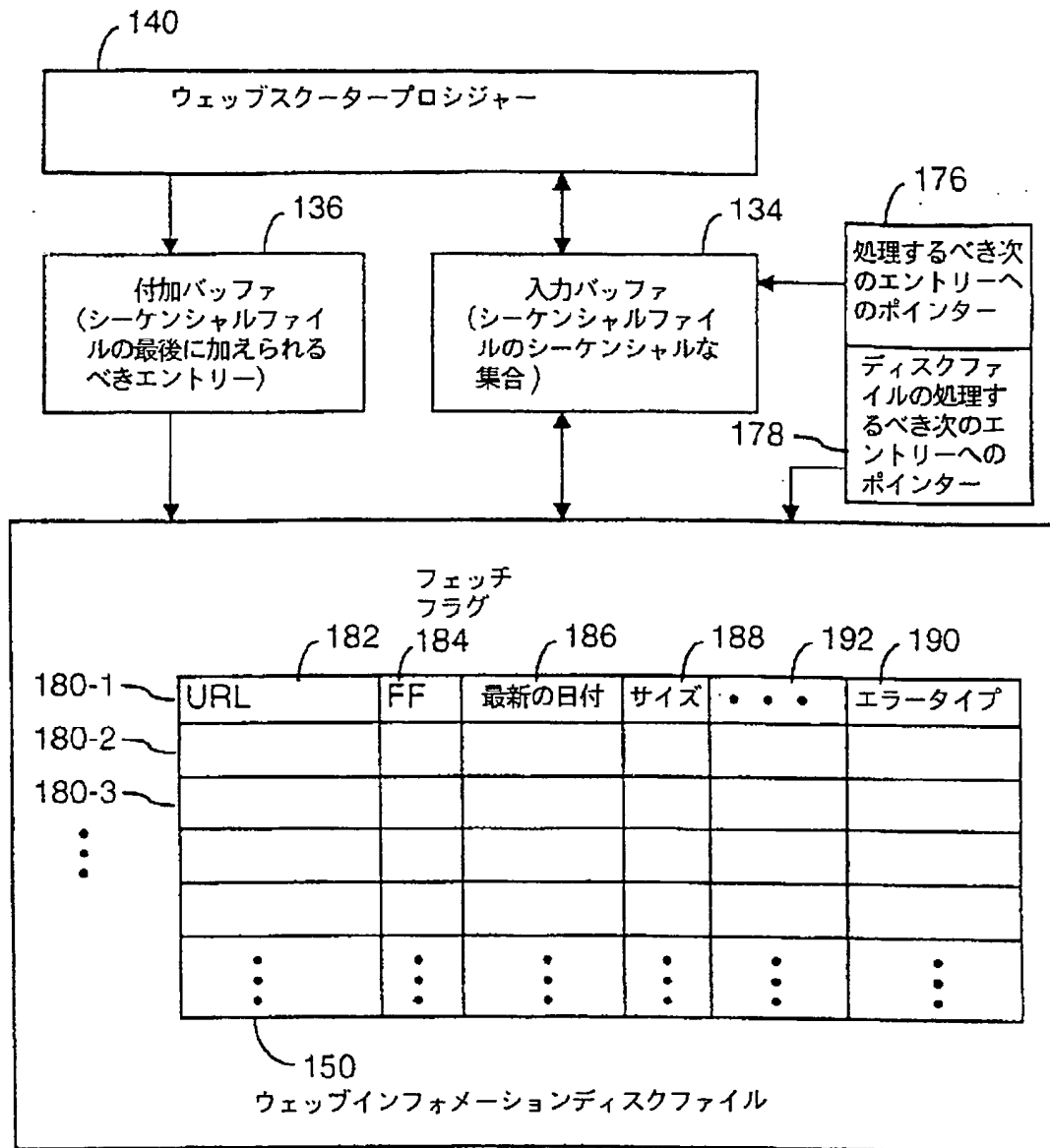


FIG. 3

【図4】

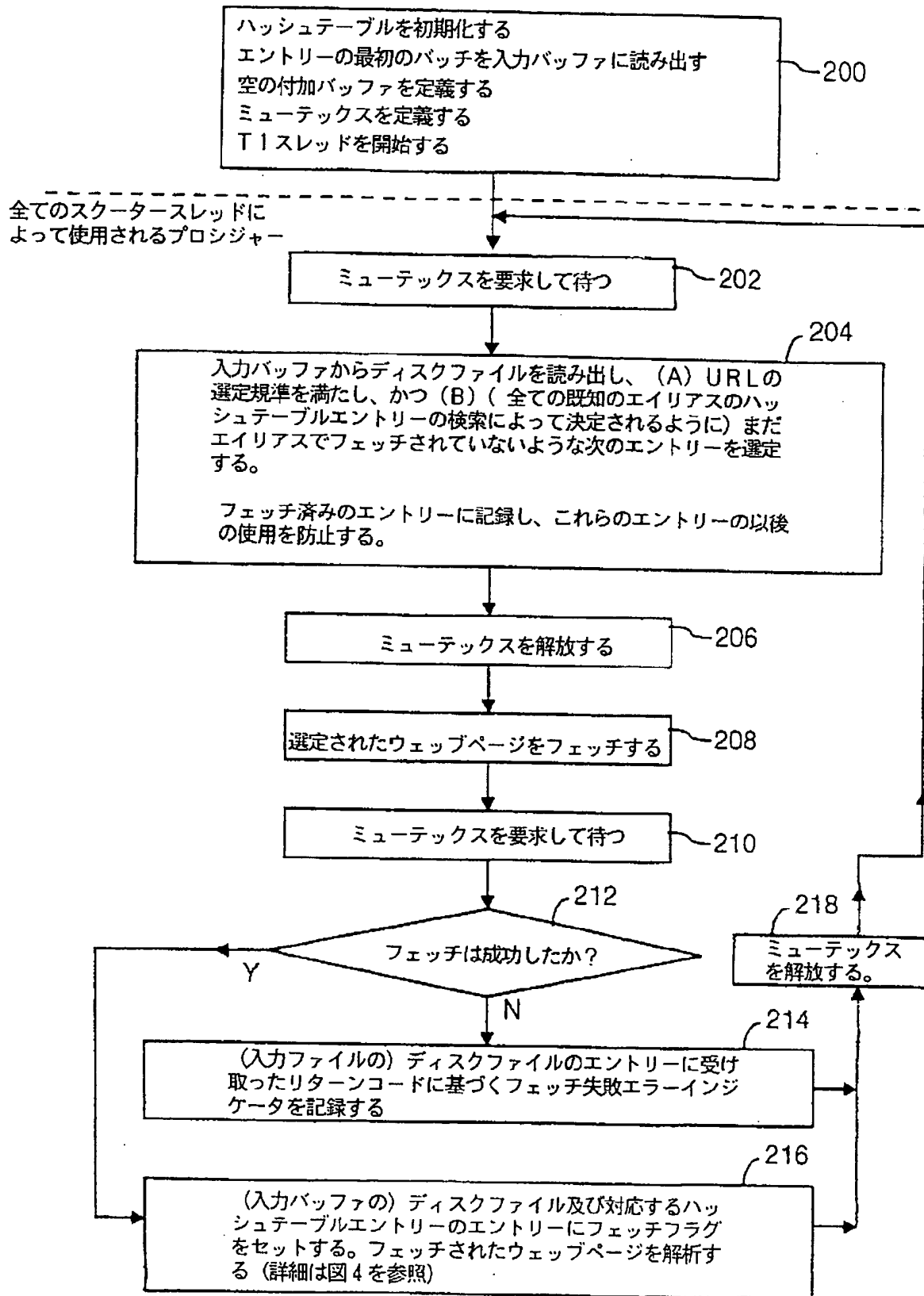


FIG. 4A

【図4】

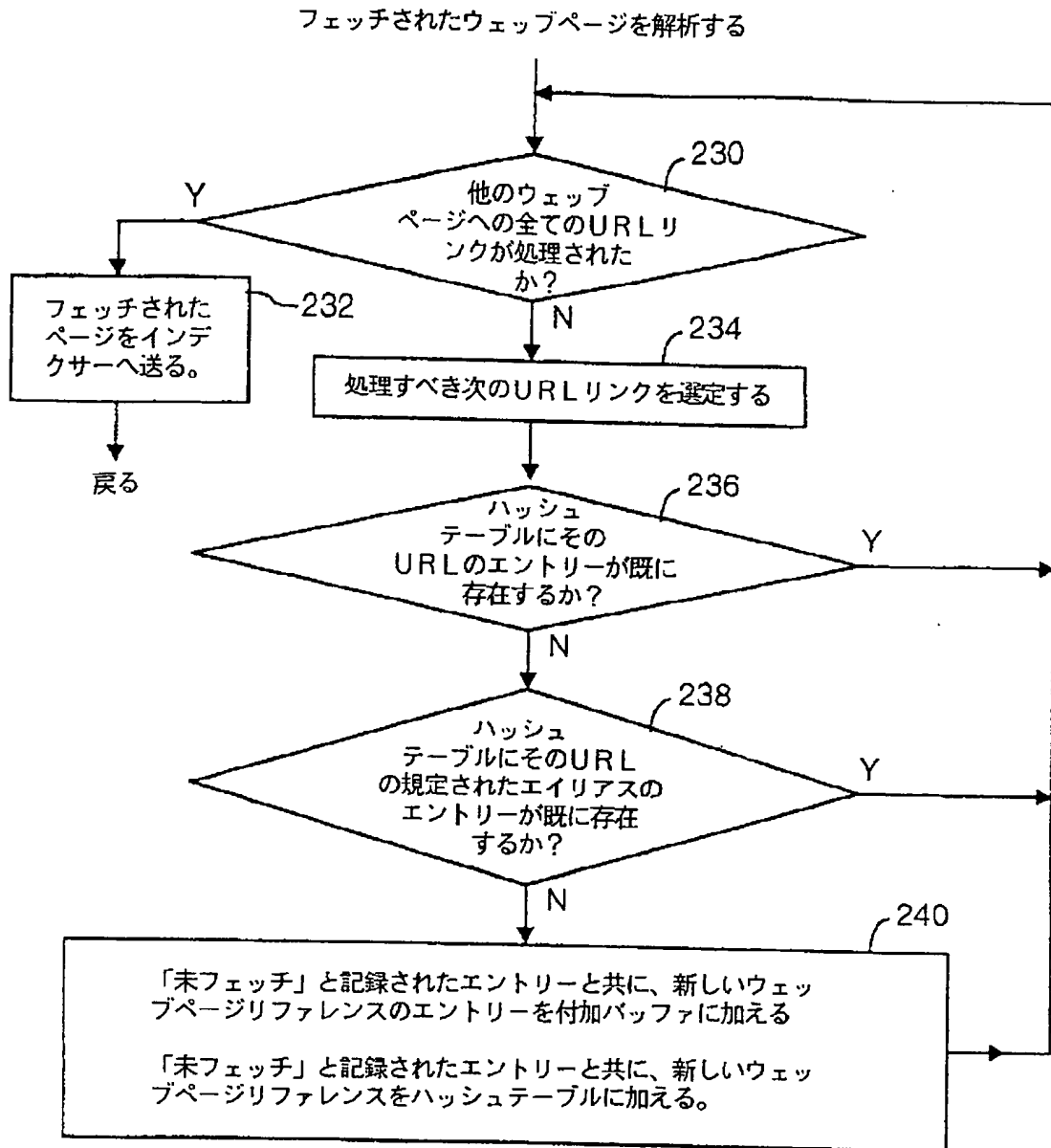


FIG. 4B

FIG. 4A

FIG. 4B

FIG. 4

【國際調查報告】

INTERNATIONAL SEARCH REPORT

A. CLASSIFICATION OF SUBJECT MATTER IPC 6 G06F17/30		International Application No PCT/US 96/19831
According to International Patent Classification (IPC) or to both national classification and IPC		
B. FIELDS SEARCHED Minimum documentation searched (classification system followed by classification symbols) IPC 6 G06F		
Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched		
Electronic data base consulted during the international search (name of data base and, where practical, search terms used)		
C. DOCUMENTS CONSIDERED TO BE RELEVANT		
Category *	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	PERSONAL COMPUTER MAGAZINE, JAN. 1996, VNU BUSINESS PUBLICATIONS, UK, pages 90-92, 94, 97 - 98, 100, XP000646762 SIMPSON D ET AL: "The searchers World Wide Web search engines" see the whole document	1, 6, 11, 16
A	ELECTRONIC LIBRARY, OCT. 1995, LEARNED INFORMATION, UK, vol. 13, no. 5, ISSN 0264-0473, pages 467-476, XP000647883 SHA V T: "Cataloguing Internet resources: the library approach" see the whole document	1, 6, 11, 16
<input type="checkbox"/> Further documents are listed in the continuation of box C.		
<input type="checkbox"/> Patent family members are listed in annex.		
* Special categories of cited documents:		
"A" document defining the general state of the art which is not considered to be of particular relevance "E" earlier document but published on or after the international filing date "L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified) "O" document referring to an oral disclosure, use, exhibition or other means "P" document published prior to the international filing date but later than the priority date claimed "T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention "X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone "Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art "&" document member of the same patent family		
Date of the actual completion of the international search 4 April 1997		Date of mailing of the international search report 16. 04.97
Name and mailing address of the ISA European Patent Office, P.B. 3818 Patentlaan 2 NL - 2280 HV Rijswijk Tel. (+31-70) 340-2040, Tx. 31 651 epo nl, Fax (+31-70) 340-3016		Authorized officer Katerbau, R

フロントページの続き

(81)指定国 EP(AT, BE, CH, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE), OA(BF, BJ, CF, CG, CI, CM, GA, GN, ML, MR, NE, SN, TD, TG), AP(KE, LS, MW, SD, SZ, UG), AM, AT, AU, BB, BG, BR, BY, CA, CH, CN, CZ, DE, DK, ES, FI, GB, GE, HU, JP, KE, KG, KP, KR, KZ, LK, LT, LU, LV, MD, MG, MN, MW, NO, NZ, PL, PT, RO, RU, SD, SE, SI, SK, TJ, TT, UA, UZ, VN

【要約の続き】

一ケンシャルに行われ、単一のI/Oオペレーションとして、シーケンシャルディスクファイルからの多数のエントリーが入力バッファへ移されるようにする。従って、シーケンシャルディスクファイルは入力バッファからアクセスされる。同様に、シーケンシャルファイルに加えられるべき全ての新しいエントリーは付加バッファに記憶され、付加バッファが一杯になった時はいつでも、付加バッファの内容はシーケンシャルファイルの最後に加えられる。このようにして、ウェブインフォメーションディスクファイルへのランダムアクセスは排除され、ディスクアクセス制限によって引き起こされる待ち時間は減少される。

**This Page is Inserted by IFW Indexing and Scanning
Operations and is not part of the Official Record**

BEST AVAILABLE IMAGES

Defective images within this document are accurate representations of the original documents submitted by the applicant.

Defects in the images include but are not limited to the items checked:

- ☐ **BLACK BORDERS**
- ☐ **IMAGE CUT OFF AT TOP, BOTTOM OR SIDES**
- ☐ **FADED TEXT OR DRAWING**
- ☒ **BLURRED OR ILLEGIBLE TEXT OR DRAWING**
- ☐ **SKEWED/SLANTED IMAGES**
- ☐ **COLOR OR BLACK AND WHITE PHOTOGRAPHS**
- ☐ **GRAY SCALE DOCUMENTS**
- ☐ **LINE(S) OR MARK(S) ON ORIGINAL DOCUMENT**
- ☐ **REFERENCE(S) OR EXHIBIT(S) SUBMITTED ARE POOR QUALITY**
- ☐ **OTHER:** _____

IMAGES ARE BEST AVAILABLE COPY.

As rescanning these documents will not correct the image problems checked, please do not report these problems to the IFW Image Problem Mailbox.